Solutions to Exercises
Chapter 4: Recurrence relations and generating functions

1 (a) There are n seating positions arranged in a line. Prove that the number of ways of choosing a subset of these positions, with no two chosen positions consecutive, is F_{n+1}.

(b) If the n positions are arranged around a circle, show that the number of choices is $F_n + F_{n-2}$ for $n \geq 2$.

(a) Proof by induction. If $g(n)$ denotes this number, then we have $g(1) = 2 = F_2$, $g(2) = 3 = F_3$. (For $n = 2$, we cannot occupy both positions; but all other choices are possible.) For $n > 2$, we separate the seating selections into those in which the last position is unoccupied and those in which it is occupied. There are $g(n-1)$ of the first kind. If the last position is taken, then the one before it must be free, and we have an arbitrary seating plan on the first $n-2$ positions; so there are $g(n-2)$ of these. Hence

$$g(n) = g(n-1) + g(n-2) = F_n + F_{n-1} = F_{n+1},$$

and the inductive step is proved.

(b) Consider a particular position on the circle. If it is unoccupied, we can break the circle at that point, and obtain a line with $n-1$ positions, which can be filled in $g(n-1) = F_n$ ways. If the position is occupied, then its neighbours on either side are unoccupied, and (if $n \geq 3$) we can remove the position and its two neighbours, obtaining a line of $n-3$ positions which can be filled in $g(n-3) = F_{n-2}$ ways. The same holds if $n = 2$, since there is just one seating plan with the given position occupied, and $F_0 = 1$. So we have $F_n + F_{n-2}$ seating plans in all.

2 Prove the following identities:

(a) $F_n^2 - F_{n+1}F_{n-1} = (-1)^n$ for $n \geq 1$.

(b) $\displaystyle\sum_{i=0}^{n} F_i = F_{n+2} - 1$.

(c) $F_{n-1}^2 + F_n^2 = F_{2n}$, $\qquad F_{n-1}F_n + F_nF_{n+1} = F_{2n+1}$.

(d) $\displaystyle F_n = \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n-i}{i}$.

(a) Induction: the result holds for $n = 1$. For $n \geq 2$, we have

$$F_n^2 - F_{n+1}F_{n-1} = F_n^2 - (F_n + F_{n-1})F_{n-1} = (F_n - F_{n-1})F_n - F_{n-1}^2 = -(F_{n-1}^2 - F_nF_{n-2}),$$

so, if $F_{n-1}^2 - F_nF_{n-2} = (-1)^{n-1}$ then $F_n^2 - F_{n+1}F_{n-1} = (-1)^n$.

(b) Induction. The result is true for $n = 0$ (the empty sum is zero). Assuming it for $n-1$, we have

$$\sum_{i=0}^{n} F_i = (F_{n+1} - 1) + F_n = F_{n+2} - 1.$$

(c) Again, induction. The result holds for $n = 1$ by inspection. Assume it for n; that is,

$$\begin{aligned} F_{n-2}^2 + F_{n-1}^2 &= F_{2n-2}, \\ F_{n-2}F_{n-1} + F_{n-1}F_n &= F_{2n-1}. \end{aligned}$$

Adding these equations and using the Fibonacci recurrence, we get

$$F_{n-2}F_n + F_{n-1}F_{n+1} = F_{2n}.$$

Using (a) twice, this implies that $F_{n-1}^2 + F_n^2 = F_{2n}$. Now add this equation to the second displayed equation using the Fibonacci recurrence to get

$$F_{n-1}F_n + F_nF_{n+1} = F_{2n+1}.$$

(d) Most easily, this follows from our original interpretation of F_n as the number of expressions for n as an ordered sum of 1s and 2s. Such an expression with i 2s will have $n - 2i$ 1s, hence $n - i$ summands altogether; there are $\binom{n-i}{i}$ ways to choose the positions of the 2s in the sum. Now summing over i gives the result.

3 Show that F_n is composite for all *odd* $n > 3$.

By 2(c), $F_{2n+1} = F_n(F_{n-1} + F_{n+1})$; and if $n > 1$, then both factors are greater than 1.

4 Show that

$$\sum_{i=0}^{\lfloor (n-1)/2 \rfloor} F_{n-2i} = F_{n+1} - 1$$

for $n \geq 1$.

The proof is by an induction which goes from $n-2$ to n, so the initial cases $n = 1$ and $n = 2$ must both be checked. Assuming the result for $n-2$, we have

$$\sum_{i=1}^{\lfloor (n-1)/2 \rfloor} F_{n-2i} = (F_{n-1} - 1) + F_n = F_{n+1} - 1.$$

5 Prove that every non-negative integer x less than F_{n+1} can be expressed in a unique way in the form

$$F_{i_1} + F_{i_2} + \ldots + F_{i_r},$$

where $i_1, i_2, \ldots, i_r \in \{1, \ldots, n\}$, $i_1 > i_2 + 1$, $i_2 > i_3 + 1$, ...(in other words, $i_1, \ldots, i_r$ are all distinct and no two are consecutive). Deduce Exercise 1(a).

Follow the hint. If we had any expression of this form using Fibonacci numbers below F_n, then we could if necessary replace the summands by larger ones and add new summands to obtain $F_{n-1} + F_{n-3} + \ldots = F_n - 1$ (by Question 4). So the sum of the original expression was at most $F_n - 1$. Hence any expression summing to x, with $F_n \leq x < F_{n+1}$, must include F_n. Now $x - F_n < F_{n+1} - F_n = F_{n-1}$, so by induction there is a unique expression for $x - F_n$, and hence for x (since the expression for $x - F_n$ cannot involve F_{n-1}, so does not contain consecutive Fibonacci numbers).

Hence, the number of expressions of this form (that is, the number of ways of choosing a subset of the indices $1, 2, \ldots, n$ with no two consecutive) is equal to the number of possible sums $0, 1, \ldots, F_{n+1} - 1$, that is, F_{n+1}, as asserted in 1(a).

6 Fibonacci numbers are traditionally associated with the breeding of rabbits. Assume that a pair of rabbits does not breed in its first month, and that it produces a pair of offspring in each subsequent month. Assume also that rabbits live forever. Show that, starting with one newborn pair of rabbits, the number of pairs alive in the n^{th} month is F_n.

In the 0th and 1st months, one pair is alive, so the result is true for $n = 0, 1$. Assume it holds for $n-1$. Then, in the $(n-1)$st month, F_{n-1} pairs of rabbits are alive, of whom F_{n-2} were alive in the preceding month (and hence old enough to breed), providing F_{n-2} newborn pairs in the nth month. Thus, the total number of pairs in the nth month is $F_{n-1} + F_{n-2} = F_n$.

7 Prove that the number of additions required to compute the Fibonacci number F_n according to the 'inefficient' algorithm described in the text is $F_n - 1$.

Induction: check the result for small n. Now F_{n-1} takes $F_n - 1$ additions, and F_{n-2} takes $F_{n-1} - 1$ additions; one further addition is required to combine them, giving in all $(F_n - 1) + (F_{n-1} - 1) + 1 = F_{n+1} - 1$ additions.

8 (a) Prove that $F_{m+n} = F_m F_n + F_{m-1} F_{n-1}$ for $m, n \geq 0$ (with the convention that $F_{-1} = 0$).

(b) Use this to derive an algorithm for calculating F_n using only $c \log n$ arithmetic operations.

(c) Given that multiplication is slower than addition, is this algorithm really better than one involving $n-1$ additions?

(a) Induction on m. For $m = 0$, the result is a tautology, and for $m = 1$ it is the Fibonacci recurrence. For $m > 1$ we have

$$\begin{aligned} F_{(m+1)+n} &= F_{m+n} + F_{(m-1)+n} \\ &= F_m F_n + F_{m-1} F_{n-1} + F_{m-1} F_n + F_{m-2} F_{n-1} \\ &= F_{m+1} F_n + F_m F_{n-1}, \end{aligned}$$

using the convention that $F_{-1} = 0$ and the fact that, then, the relation $F_m = F_{m-1} + F_{m-2}$ holds also for $m = 1$.

(b) The trick is to calculate pairs (F_{n-1}, F_n) of Fibonacci numbers, observing that we can go efficiently from the pairs (F_{m-1}, F_m) and (F_{n-1}, F_n) to the pair (F_{m+n-1}, F_{m+n}) using (a). A cleaner way to express this is in terms of matrices. We have $(F_n \; F_{n+1})A = (F_{n+1} \; F_{n+2})$, where A is the 2×2 matrix $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$. Hence $(F_n \; F_{n+1}) = (1, 1)A^n$. By the analogue for powers of Russian peasant multiplication (Chapter 2, Exercise 12(iii)), we can find A^n in about $2 \log_2 n$ matrix multiplications, each requiring eight integer multiplications and four additions (though this can be improved).

(c) Since F_n is exponentially large, the number of its digits is proportional to n. Hence a long multiplication involves a number of additions proportional to n, and we have not really made much saving over a method involving n additions.

9 (a) Solve the following recurrence relations.

(i) $f(n+1) = f(n)^2$, $f(0) = 2$.

(ii) $f(n+1) = f(n) + f(n-1) + f(n-2)$, $f(0) = f(1) = f(2) = 1$.

(iii) $f(n+1) = 1 + \sum_{i=0}^{n-1} f(i)$, $f(0) = 1$.

(b) Show that the number of ways of writing n as a sum of positive integers, where the order of the summands is significant, is 2^{n-1} for $n \geq 1$.

(a) (i) Let $g(n) = \log_2 f(n)$. Then $g(n+1) = 2g(n)$, $g(0) = 1$; so $g(n) = 2^n$ (Section 3.1), and $f(n) = 2^{2^n}$.

(ii) The characteristic equation (see Section 4.3) is $x^3 = x^2 + x + 1$. This cubic has three distinct real roots α, β, γ, and so the general solution is $f(n) = a\alpha^n + b\beta^n + c\gamma^n$, where a, b, c are determined by the three equations $a + b + c = 1$, $a\alpha + b\beta + c\gamma = 1$, $a\alpha^2 + b\beta^2 + c\gamma^2 = 1$.

(iii) We have $f(1) = 1$, since the empty sum is zero. For $n \geq 1$, we have

$$f(n+1) = 1 + \sum_{i=0}^{n-1} f(i) = \left(1 + \sum_{i=0}^{n-2} f(i)\right) + f(n-1) = f(n) + f(n-1),$$

and so by induction $f(n)$ is the Fibonacci number F_n.

(b) There is one expression consisting of the single number n. Any other expression ends with some positive number $j < n$, preceded by an expression with sum $n - j$; so the number $f(n)$ of expressions satisfies

$$f(n) = 1 + \sum_{j=1}^{n-1} f(n-j) = 1 + \sum_{i=1}^{n-1} f(i).$$

Just as in (a)(iii), we find that $f(n+1) = f(n) + f(n) = 2f(n)$. Since $f(1) = 1$, we have $f(n) = 2^{n-1}$, as claimed.

10 The number $f(n)$ of steps required to solve the 'Chinese rings puzzle' with n rings satisfies $f(1) = 1$ and

$$f(n+1) = \begin{cases} 2f(n), & n \text{ odd}, \\ 2f(n)+1, & n \text{ even}. \end{cases}$$

Prove that $f(n+2) = f(n+1) + 2f(n) + 1$. Hence or otherwise find a formula for $f(n)$.

If n is odd, then $f(n+2) = 2f(n+1)+1$ and $f(n+1) = 2f(n)$; if n is even, then $f(n+2) = 2f(n+1)$ and $f(n+1) = 2f(n)+1$. So the result holds in either case.

This is not a "pure" recurrence relation because of the added 1. But if we set $g(n) = f(n) + \frac{1}{2}$, then $g(n+1) + 2g(n) = f(n+1) + 2f(n) + \frac{3}{2} = g(n+2)$. The characteristic equation is $x^2 = x + 2$, with solutions 2 and -1, so $g(n) = a2^n + b(-1)^n$. The initial values $g(1) = \frac{3}{2}$ and $g(2) = \frac{5}{2}$ yield $a = \frac{2}{3}$ and $b = -\frac{1}{6}$. So $f(n) = \frac{2}{3}2^n - \frac{1}{6}(-1)^n - \frac{1}{2}$.

11 (a) Let $s(n)$ be the number of sequences $(x_1, \ldots, x_k)$ of integers satisfying $1 \le x_i \le n$ for all i and $x_{i+1} \ge 2x_i$ for $i = 1, \ldots, k-1$. (The length of the sequence is not specified; in particular, the empty sequence is included.) Prove the recurrence

$$s(n) = s(n-1) + s(\lfloor n/2 \rfloor)$$

for $n \ge 1$, with $s(0) = 1$. Calculate a few values of s. Show that the generating function $S(t)$ satisfies $(1-t)S(t) = (1+t)S(t^2)$.

(b) Let $u(n)$ be the number of sequences $(x_1, \ldots, x_k)$ of integers satisfying $1 \le x_i \le n$ for all i and $x_{i+1} > \sum_{j=1}^{i} x_j$ for $i = 1, \ldots, k-1$. Calculate a few values of u. Can you discover a relationship between s and u? Can you prove it?

(a) We have $s(0) = 1$, since only the empty sequence occurs. For $n > 0$, divide the sequences counted by $s(n)$ into two classes: those not containing n, and those containing n. There are $s(n-1)$ of the first class. For the second, all terms other than n are at most $n/2$, so such a sequence is obtained from a sequence of numbers with $1 \le x_i \le \lfloor n/2 \rfloor$ and $x_{i+1} \ge 2x_i$ by adjoining n; so there are $s(\lfloor n/2 \rfloor)$ of these. The recurrence relation follows.

If $S(t) = \sum s(n)t^n$, then we have

$$S(t) = tS(t) + (1+t)S(t^2);$$

for the coefficient of t^n on the right is $s(n-1)+s(\lfloor n/2 \rfloor)$, and the constant term on both sides is equal to 1.

(b) The relationship is $u(n)-u(n-1)=s(n)/2$ for $n \geq 2$. The proof is quite intricate, and depends on defining another sequence $v(n)$ as follows: $v(n)$ is the number of sequences $(x_1,\ldots,x_k)$ of positive integers which satisfy $x_i > \sum_{j=1}^{i-1} x_j$ for all $i \leq k$ and $\sum_{i=1}^{k} x_i = n$. Now such a sequence must have last term $x_k > n/2$, and is obtained by taking a sequence with sum $n-x_k \leq \lfloor (n-1)/2 \rfloor$ and appending x_k to it. So we have the recurrence

$$v(n) = \sum_{i=1}^{\lfloor (n-1)/2 \rfloor} v(i).$$

It follows from this that $v(2n) = v(2n-1) = v(2n-2)+v(n-1)$ for all $n > 0$.

Now we claim that $v(2n) = s(n)/2$ for all $n \geq 1$. This is proved by induction, being true for $n = 1$. Assuming the result for $n-1$, we have

$$\begin{aligned} v(2n) &= v(2n-2)+v(n-1) \\ &= s(n-1)/2+s(\lfloor n/2 \rfloor)/2 \\ &= s(n)/2, \end{aligned}$$

and the induction goes through.

Now $u(n)-u(n-1)$ is the number of sequences satisfying the specification for u which have last term n. These are obtained by appending n to a sequence with sum strictly smaller than n; so we have

$$\begin{aligned} u(n)-u(n-1) &= \sum_{i=0}^{n-1} v(i) \\ &= v(2n) \\ &= s(n)/2. \end{aligned}$$

Remark: Sequences satisfying this condition are called *superincreasing*; they arose in the first example of a public-key cryptosystem, the Merkle–Hellman knapsack system.

The two sequences s and u occur as numbers M1011 and M1053 in the *Encyclopedia of Integer Sequences*, where further references can be found, or you can find them in the online version here (for s) and here (for u).

12 Let $F(t)$ be a formal power series with constant term 1. By finding a recurrence relation for its coefficients, show that there is a multiplicative inverse $G(t)$ of $F(t)$. Moreover, if the coefficients of F are integers, so are those of G.

Let $F(t) = \sum f_n t^n$ and $G(t) = \sum g_n t^n$, with $f_0 = 1$. The relation $F(t)G(t) = 1$ is equivalent to $g_0 = 1$ and the recurrence relation

$$g_n = -\sum_{i=1}^{n} f_i g_{n-i}$$

for g_n in terms of earlier values. So the values g_n are determined, and are integers if the values f_n are.

13 A permutation π of the set $\{1,\ldots,n\}$ is called *connected* if there does not exist a number k with $1 \leq k < n$ such that π maps the subset $\{1,2,\ldots,k\}$ into itself. Let c_n be the number of connected permutations. Prove that

$$\sum_{i=1}^{n} c_i (n-i)! = n!$$

Deduce that, if $F(t) = \sum_{n\geq 1} n! t^n$ and $G(t) = \sum_{n\geq 1} c_n t^n$ are the generating functions of the sequences $(n!)$ and (c_n) respectively, then $1 - G(t) = (1+F(t))^{-1}$.

For a permutation π, let $i(\pi)$ be the least positive integer k such that π maps the set $\{1,\ldots,k\}$ into itself. (Such a k exists, since certainly π maps $\{1,\ldots,n\}$ to itself.) Then π is the composite of a connected permutation on $\{1,\ldots,i(\pi)\}$ and an arbitrary permutation on $\{i(\pi)+1,\ldots,n\}$ (this last set being empty if $i(\pi) = n$). Summing over i, we obtain the stated recurrence.

As in the preceding question, this recurrence is equivalent to $(1+F(t))(1-G(t)) = 1$.

14 Let

$$\prod_{n\geq 1} (1+t^n) = \sum_{n\geq 0} a_n t^n.$$

Prove that a_n is the number of ways of writing n as the sum of *distinct* positive integers. (For example, $a_6 = 4$, since $6 = 5+1 = 4+2 = 3+2+1$.)

A term in the infinite product $\prod(1+t^n)$ is obtained by selecting t^m from the mth factor for a finite number of values of n, and 1 from all the other factors. So there is a contribution of t^n for every expression of n as a sum of distinct positive integers.

15 (a) In an election, there are two candidates, A and B; the number of votes cast is $2n$. Each candidate receives exactly n votes; but, at every intermediate point during the count, A has received more votes than B. Show that the number of ways this can happen is the Catalan number C_n. Can you construct a bijection between the bracketed expressions and the voting patterns in (a)?

(b) In the above election, assume only that, at any intermediate stage, A has received at least as many votes as B. Prove that the number of possibilities is now C_{n+1}.

(a) As in the Hint, let $f(n)$ be the number of ways of counting. Now A leads by just one vote after the first vote is counted. Suppose that this next occurs after $2i+1$ votes have been counted. (It must happen again, since B eventually catches A.) Then there are $f(i)$ choices for the count between these points, since each candidate receives i votes and A is always strictly ahead. Also, there are $f(n-i)$ choices for the rest of the count; for if we pretend at this stage that A has just one vote and B none, we are running the count satisfying the same conditions with just $2(n-i)$ votes altogether. So we have $f(n) = \sum_{i=1}^{n-1} f(i)f(n-i)$, which is the Catalan recurrence.

(b) Again follow the hint: In the modified election where A gets an extra vote at the start and B an extra vote at the end, there are $n+1$ votes for each candidate, and A is alway strictly ahead. So there are C_{n+1} ways of doing the count under this condition.

16 A clown stands on the edge of a swimming pool, holding a bag containing n red and n blue balls. He draws the balls out one at a time and discards them. If he draws a blue ball, he takes one step back; if a red ball, one step forward. (All steps have the same size.) Show that the probability that the clown remains dry is $1/(n+1)$.

Imagine that the clown wears a diving suit, and continues to draw balls even after he gets wet. There are $\binom{2n}{n}$ ways in which the balls could be drawn. The clown stays dry if and only if the number of red balls never exceeds the number of blue ones; according to the voting interpretation, this is C_{n+1}. The ratio is $1/(n+1)$.

For a harder exercise, find a direct proof of this exercise, and reverse the above argument to deduce the formula for the Catalan numbers.

17 Prove that

$$\lim_{n\to\infty} \left(\frac{B_n}{n!}\right)^{1/n} = 0.$$

The radius of convergence of the power series $\sum_{n\geq 0} B_n t^n/n!$ is the reciprocal of

$$\limsup_{n\to\infty} \left(\frac{B_n}{n!}\right)^{1/n}.$$

But the radius of convergence is infinite, since the series converges everywhere. (In general, the radius of convergence is the distance from the origin to the nearest singularity in the complex plane; if the sum function has no singularities, the radius of convergence is infinite.) So the lim sup is zero, which means (since all the values are positive) that the limit is zero.

18 (a) Prove that the exponential generating function for the number $s(n)$ of involutions on $\{1,\ldots,n\}$ (Section 4.4) is $\exp(t+\frac{1}{2}t^2)$.

(b) Prove that the exponential generating function for the number $d(n)$ of derangements of $\{1,\ldots,n\}$ is $1/((1-t)\exp(t))$.

(a) The function $S(t) = \exp(t+\frac{1}{2}t^2)$ is the unique solution of the differential equation $S'(t) = (1+t)S(t)$ with the initial condition $S(0) = 1$. So it suffices to check that the e.g.f. of the numbers $s(n)$ satisfies this differential equation. (Clearly it satisfies the initial condition.) Now, if we put $S(t) = \sum_{n\geq 0} s(n)t^n/n!$, then the coefficient of $t^{n-1}/(n-1)!$ in $(1+t)S(t)$ is $s(n-1)+(n-1)s(n-2)$, whereas the coefficient in $S'(t)$ is $s(n)$ (since differentiation of the e.g.f. corresponds to a left shift of the sequence). By the recurrence relation in Section 4.4, these two expressions are equal.

(b) We have

$$d(n) = n!\left(\sum_{i=0}^{n} \frac{(-1)^i}{i!}\right).$$

by (4.4.1). Hence

$$\begin{aligned}\sum_{n\geq 0} \frac{d(n)t^n}{n!} &= \left(\sum_{i\geq 0} \frac{(-t)^i}{i!}\right)\cdot\left(\sum_{k\geq 0} t^k\right) \\ &= \frac{1}{\mathrm{e}^t(1-t)}.\end{aligned}$$

(In the first line, we put $k = n-i$; then i and k run independently over the non-negative integers.)

19 The *Bernoulli numbers* $B(n)$ (not to be confused with the Bell numbers B_n) are defined by the recurrence $B(0) = 1$ and

$$\sum_{k=0}^{n} \binom{n+1}{k} B(k) = 0$$

for $n \geq 1$. Prove that the exponential generating function

$$f(t) = \sum_{n \geq 0} \frac{B(n)t^n}{n!}$$

is given by $f(t) = t/(\exp(t) - 1)$.

Show that $f(t) + \frac{1}{2}t$ is an even function of t, and deduce that $B(n) = 0$ for all odd $n \geq 3$.

What is the solution of the similar-looking recurrence $b(0) = 1$ and

$$\sum_{k=0}^{n} \binom{n}{k} b(k) = 0$$

for $n \geq 1$?

The recurrence can be written as

$$\sum_{k=1}^{n+1} \binom{n+1}{k} B(k) = B(n+1) \qquad \text{for } n \geq 2.$$

Multiplying by $t^{n+1}/(n+1)!$ and summing over $n \geq 2$, the two sides are $f(t)\exp(t)$ and $f(t)$ with the constant and linear terms omitted. Thus $f(t)\exp(t) - 1 - t + \frac{1}{2}t = f(t) - 1 + \frac{1}{2}t$, whence $f(t) = t/(\exp(t) - 1)$, as required.

Now $f(t) + \frac{1}{2}t = \frac{1}{2}t(\exp(\frac{1}{2}t) + \exp(-\frac{1}{2}t))/(\exp(\frac{1}{2}t) - \exp(-\frac{1}{2}t)) = \frac{1}{2}t\coth\frac{1}{2}t$, an even function.

The last recurrence obviously has the solution $b(k) = (-1)^k$.

20 For even n, let e_n be the number of permutations of $\{1,\ldots,n\}$ with all cycles even; o_n, the number of permutations with all cycles odd; and $p_n = n!$ the total number of permutations. Let $E(t)$, $O(t)$ and $P(t)$ be the exponential generating functions of these sequences. Show that

(a) $P(t) = (1-t^2)^{-1}$;

(b) $E(t) = (1-t^2)^{-1/2}$; [HINT: Exercise 15 of Chapter 3]

(c) $E(t).O(t) = P(t)$;

(d) $e_n = o_n$ for all even n.

Find a 'bijective' proof of the last equality.

(a) Since we are only considering even vales of n, we have

$$P(t) = \sum_{m\geq 0} \left(\frac{(2m)!}{(2m)!}\right) t^{2m} = \frac{1}{1-t^2}.$$

(b) By Chapter 3, Exercise 16, the number e_{2m} of permutations of a $(2m)$-set with all cycles even is $((2m)!!)^2$. So we have

$$E(t) = \sum_{m\geq 0} \left(\frac{((2m)!!)^2}{(2m)!}\right) t^{2m} = (1-t^2)^{-1/2},$$

the second equality being verified by expanding the last expression using the Binomial Theorem.

(c) This is an instance of a very general counting principle. If $F(t)$ and $G(t)$ are the e.g.f.s for labelled structures in two classes $\mathcal{F}$ and $\mathcal{G}$, with $F(t) = \sum f_n t^n/n!$ and $G(t) = \sum g_n t^n/n!$, then $F(t)G(t)$ is the e.g.f. of structures consisting of a partition of the point set with an $\mathcal{F}$-structure on one part and a $\mathcal{G}$-structure on the complement. For, if $\mathcal{H}$ is the class of such composite structures, then the number of $\mathcal{H}$-structures on an n-set is given by

$$h_n = \sum_{k=0}^{n} \binom{n}{k} f_k g_{n-k},$$

from which a simple calculation shows that $H(t) = \sum h_n t^n/n! = F(t)G(t)$.

Now (c) follows on applying this result, where $\mathcal{F}$ and $\mathcal{G}$ denote the classes of permutations with all cycles even, resp., all cycles odd: given an arbitrary

permutation π on $\{1,\ldots,2m\}$, there is a unique partition of the set into two parts on which the induced permutations satisfy these conditions; and both parts have even cardinality.

(d) From (a), (b) and (c), we deduce that $O(t) = (1-t^2)^{-1/2} = E(t)$; so $o_n = e_n$ for all n.

Here is an outline of the construction of a bijection: you should fill in the details. Given the cycle decomposition of a permutation of $\{1,\ldots,n\}$, we can rotate each cycle so that the least element comes first, and then order the cycles according to their least elements. Call this the *canonical form* of the cycle decomposition. We now define the bijections:

(a) Given a canonical form $C_1,\ldots,C_{2r}$, where each cycle C_i has odd length, for $i = 1,\ldots,r$ remove the last element of the cycle C_{2i} and add it at the end of the cycle C_{2i-1}. (Some cycles may disappear in this process.) The resulting cycles all have even length, and the expression is in the canonical form.

(b) Given a canonical form $C_1,\ldots,C_m$, where each cycle has even length, proceed recursively as follows. Let x be the last element of C_1, and y the first element of C_2. (Take $y = \infty$ if C_2 doesn't exist, that is, if there is only one cycle.)

- If $x > y$, remove x from C_1 and add it at the end of C_2; then process $C_3,\ldots,C_m$.
- If $x < y$, remove x from C_1 and add it as a new singleton cycle after C_1; then process $C_2,\ldots,C_m$.

The resulting cycles all have odd length and the expression is in canonical form.

It remains to show that these two maps are mutually inverse. See R. P. Lewis and S. P. Norton, *Discrete Mathematics* **138** (1995), 315–318, for further details.

21 Show that QUICKSORT sometimes requires all $\binom{n}{2}$ comparisons to sort a list. For how many orderings does this occur? One such ordering is the case when the list is already sorted — is this a serious defect of QUICKSORT?

Suppose that we start with a sorted list. Then QUICKSORT selects the first element (which is the smallest), and partitions the remainder into the empty list and all the other elements (with $n-1$ comparisons). One branch of the recursion is trivial, but in the other we again have a sorted list, requiring (by induction) $\binom{n-1}{2}$ comparisons. So $\binom{n}{2}$ comparisons are needed altogether.

The sorted list is not the only permutation requiring so many comparisons. As long as each element is either smaller than or greater than all its successors, the same uneven split will occur at eac stage. So there are 2^{n-1} "bad" permutations, since there are two choices for each element except the last.

This is a problem, but in practice not too serious. If the lists to be sorted are truly random, then the proportion of bad permutations decreases faster than exponentially. If we are likely to meet sorted or partially sorted lists quite often, we can modify the algorithm by choosing a to be the middle item of the list, rather than the first.

22 Let m_n be the minimum number of comparisons required by QUICKSORT to sort a list of length n. Prove that, for each integer $k > 1$, m_n is a linear function of n on the interval from $2^{k-1}-1$ to 2^k-1, with

$$m_{2^k-1} = (k-2)2^k + 2.$$

If $n = 2^k - 1$, what can you say about the number of orderings requiring m_n comparisons?

The minimum number of comparisons will occur when the sublists are as nearly equal as possible: both of length $(n-1)/2$ if n is odd, or of lengths $(n-2)/2$ and $n/2$ if n is even. So we have the recurrence

$$m_n = \begin{cases} n-1+2m_{(n-1)/2}, & \text{if } n \text{ is odd,} \\ n-1+m_{(n-2)/2}+m_{n/2}, & \text{if } n \text{ is even.} \end{cases}$$

Suppose that $m_n = an+b$ for $c \le n \le d$. Then, for $2c+1 \le n \le 2d+1$, we have $m_n = n-1+2(a(n-1)/2+b) = (a+1)n+(2b-a-1)$. But m_n is linear (in fact identically zero) for $0 \le n \le 1$; by induction on k, it is linear on each interval $2^{k-1}-1 \le n \le 2^k-1$. If the value on this interval is given by $m_n = a_k n + b_k$, then we have $a_0 = b_0 = 0$, $a_{k+1} = a_k + 1$, and $b_{k+1} = 2b_k - a_k - 1$. By induction, we find that $a_k = k$ and $b_k = -2^{k+1} + k + 2$. Setting $n = 2^k - 1$, we have

$$m_{2^k-1} = k \cdot (2^k-1) - 2^{k+1} + k + 2 = (k-2)2^k + 2,$$

as required.

To count the number of orders which require the minimum number of comparisons, note that the first step in the algorithm splits the list into sublists of length i and $n-i-1$, say; now we require that each of these sublists can be sorted with the minimum number of comparisons, and also that both i and $n-1-i$ lie in an interval in which m_j is a linear function of j, that is, one of the form $[2^{d-1}-1, 2^d-1]$. However, the two sublists can be merged arbitrarily. So the number $f(n)$ of orders requiring the minimum number of comparisons satisfies the recurrence

$$f(n) = \sum \binom{n-1}{i} f(i) f(n-i-1),$$

where the sum is over all i such that i and $n-i-1$ lie in the same interval of the form $[2^{d-1}-1, 2^d-1]$.

For example, if $n=5$ and the list contains $1,\ldots,5$ in some order, then it requires the minimum number 6 of comparisons to sort if and only if one of the following holds:

- the first element is 3;
- the first element is 2, and 4 precedes 3 and 5;
- the first element is 4, and 2 precedes 1 and 3.

There are 40 such lists, out of 120. As in the previous question, the maximum number of comparisons is 10, and 16 lists require this number. Check that the numbers requiring $7,8,9$ comparisons are 32, 24, 8 respectively. Now check that the average agrees with the value calculated in the recurrence of Section 4.7.

23 This exercise justifies the 'Twenty Questions' principle. We are given N objects and required to distinguish them by asking questions, each of which has two possible answers. The aim of this exercise is to show that, no matter what scheme of questioning is adopted, on average the number of questions required is at least $\log_2 N$. (For some schemes, the average may be much larger. If we ask 'Is it a_1?', 'Is it a_2?', etc., then on average $(N+1)/2$ questions are needed!)

A *binary tree* is a graph (see Chapter 2) with the following properties:

- there is a vertex (the *root*) lying on just two edges;
- every other vertex lies on one or three edges (and is called a *leaf* or an *internal vertex* accordingly);
- there are no circuits (closed paths of distinct vertices), and every vertex can be reached by a path from the root.

It is convenient to arrange the vertices of the tree on successive *levels*, with the root on level 0. Then any non-leaf is joined to two *successors* on the next level, and every vertex except the root has one *predecessor*. The *height* of a vertex is the number of the level on which it lies.

In our situation, a vertex is any set of objects which can be distinguished by some sequence of questions. The root corresponds to the whole set (before any questions are asked), and leaves are singleton sets. The two successors of a vertex are the sets distinguished by the two possible answers to the next question. The height of a leaf is the number of questions required to identify that object uniquely.

STEP 1. Show that there are two leaves of maximal height (h, say) with the same predecessor. Deduce that, if there is a leaf of height less than $h-1$, we can find another binary tree with N leaves having smaller average height. Hence conclude that, in a tree with minimum average height, every leaf has height m or $m+1$, for some m.

STEP 2. Since there are no leaves at height less than m, there are altogether 2^m vertices on level m.

STEP 3. If there are p internal vertices on level m, show that there are $2p$ leaves of height $m+1$, and $N-2p = 2^m - p$ of height m; so $N = 2^m + p$, where $0 \leq p < 2^m$.

STEP 4. Prove that $\log_2(2^m+p) \leq m + 2p/(2^m+p)$, and deduce that the average height of leaves is at least $\log_2 N$.

Follow the suggested proof.

Step 1: If v is a vertex of maximum height and w its predecessor, then the other successor v' of w would also be a leaf (else its own successors would be higher than v). If there is a leaf x with height less than $h-1$, then remove the two leaves v, v' of height h and add two new successors y, y' of x with height less than h, reducing the average height without changing the number of leaves. So, in a tree of minimum height, no leaf has height less than $h-1$ (where h is the maximum height). Put $m = h-1$.

Step 2: If there are no leaves on level k, then each vertex on this level has two successors, and so the number of vertices on level $k+1$ is twice the number on level k. In our case, this holds for $k = 0, 1, \ldots, m-1$, so there are 2^m vertices on level m.

Step 3: Suppose that there are p internal vertices on level m. Then there are $2p$ vertices on level $m+1$, all of them leaves; and there are $2^m - p$ leaves on level m, giving $2^m + p$ altogether. Thus, if N is the number of leaves, then m and p are determined by N: 2^m is the largest power of 2 not exceeding N, and $p = N = 2^m$. (If N is a power of 2, we can take it to be 2^m rather than 2^{m+1}, to simplify things.)

Step 4: The average height of the leaves is

$$\frac{(2^m - p)m + 2p(m+1)}{2^m + p} = m + \frac{2p}{2^m + p}.$$

By calculus, show that $-\log_2(1-x) \leq 2x$ for $0 \leq x \leq \frac{1}{2}$. (The two expressions are equal when $x = 0$ and when $x = \frac{1}{2}$; and their difference has a unique stationary value in the interval, at $x = 1 - 1/(2\log 2)$, which is a minimum.) Apply this inequality with $x = p/(2^m + p)$, so that $m - \log_2(1-x) = m + \log_2(1 + p/2^m) = \log_2(2^m + p)$. Thus, $\log_2 N \leq m + 2p/(2^m + p)$, and we are done.

24 Suppose that the two successors of each non-leaf node in a binary tree are distinguished as 'left' and 'right'. Show that, with this convention, the number of binary trees with n leaves is the Catalan number C_n.

Let T_n be the number of binary trees with n leaves, with the left-right distinction. Then $T_1 = 1$ (the root is the unique leaf); and, for $n > 1$, a tree with n leaves is specified by a left subtree with k leaves and a right subtree with $n-k$ leaves, where $1 \leq k \leq n-1$. Hence $T_n = \sum_{k=1}^{n-1} T_k T_{n-k}$ for $n > 1$. By (4.5.1) and induction, $T_n = C_n$ for all n.